Parte 5 - Ejercicios con POO + Archivos

Aplicación Nº 19 (Auto)

Realizar una clase llamada **"Auto"** que posea los siguientes atributos

privados: _color (String)
_precio (Double)
_marca (String).
_fecha (DateTime)

Realizar un constructor capaz de poder instanciar objetos pasándole como parámetros: i. La marca y el color.
ii. La marca, color y el precio.
iii. La marca, color, precio y fecha.

Realizar un método de **instancia** llamado "**AgregarImpuestos",** que recibirá un doble por parámetro y que se sumará al precio del objeto.

Realizar un método de **clase** llamado **"MostrarAuto",** que recibirá un objeto de tipo **"Auto"** por parámetro y que mostrará todos los atributos de dicho objeto.

Crear el método de instancia **"Equals"** que permita comparar dos objetos de tipo **"Auto"**. Sólo devolverá TRUE si ambos **"Autos"** son de la misma marca.

Crear un método de clase, llamado **"Add"** que permita sumar dos objetos **"Auto"** (sólo si son de la misma marca, y del mismo color, de lo contrario informarlo) y que retorne un **Double** con la suma de los precios o cero si no se pudo realizar la operación.

Ejemplo: $importeDouble = Auto::Add($autoUno, $autoDos);

Crear un método de clase para poder hacer el alta de un Auto, guardando los datos en un archivo autos.csv.

Hacer los métodos necesarios en la clase Auto para poder leer el listado desde el archivo autos.csv

Se deben cargar los datos en un array de autos.

En testAuto.php:

- Crear **dos** objetos **"Auto"** de la misma marca y distinto color.
- Crear **dos** objetos **"Auto"** de la misma marca, mismo color y distinto precio. ● Crear **un** objeto **"Auto"** utilizando la sobrecarga restante.
- Utilizar el método **"AgregarImpuesto"** en los últimos tres objetos, agregando $ 1500 al atributo precio.
- Obtener el importe sumado del primer objeto **"Auto"** más el segundo y mostrar el resultado obtenido.
- Comparar el primer **"Auto"** con el segundo y quinto objeto e informar si son iguales o no.
- Utilizar el método de clase "**MostrarAuto**" para mostrar cada los objetos impares (1, 3, 5)

Aplicación Nº 20 (Auto - Garage)

Crear la clase **Garage** que posea como atributos privados:

_razonSocial (String)

_precioPorHora (Double)

_autos (Autos[], reutilizar la clase Auto del ejercicio anterior)

Realizar un constructor capaz de poder instanciar objetos pasándole como parámetros: i. La razón social.

ii. La razón social, y el precio por hora.

Realizar un método de **instancia** llamado **"MostrarGarage",** que no recibirá parámetros y que mostrará todos los atributos del objeto.

Crear el método de instancia **"Equals"** que permita comparar al objeto de tipo **Garaje** con un objeto de tipo **Auto.** Sólo devolverá TRUE si el auto está en el garaje.

Crear el método de instancia **"Add"** para que permita sumar un objeto **"Auto"** al **"Garage"** (sólo si el auto **no** está en el garaje, de lo contrario informarlo).

Ejemplo: $miGarage->Add($autoUno);

Crear el método de instancia **"Remove"** para que permita quitar un objeto **"Auto"** del **"Garage"** (sólo si el auto **está** en el garaje, de lo contrario informarlo). Ejemplo: $miGarage->Remove($autoUno);

Crear un método de clase para poder hacer el alta de un Garage y, guardando los datos en un archivo garages.csv.

Hacer los métodos necesarios en la clase Garage para poder leer el listado desde el archivo garage.csv

Se deben cargar los datos en un array de garage.

En testGarage.php, crear autos y un garage. Probar el buen funcionamiento de todos los métodos.

Aplicación Nº 20 BIS (Registro CSV)

Archivo: registro.php

método:POST

Recibe los datos del usuario(nombre, clave,mail)por POST ,

crear un objeto y utilizar sus métodos para poder hacer el alta,

guardando los datos en usuarios.csv.

retorna si se pudo agregar o no.

Cada usuario se agrega en un renglón diferente al anterior.

Hacer los métodos necesarios en la clase usuario

Aplicación Nº 21 (Listado CSV y array de usuarios)

Archivo: listado.php

método:GET

Recibe qué listado va a retornar(ej:usuarios,productos,vehículos,...etc),por ahora solo tenemos usuarios).

En el caso de usuarios carga los datos del archivo usuarios.csv.

se deben cargar los datos en un array de usuarios.

Retorna los datos que contiene ese array en una lista

```
<ul>
    <li>Coffee</li>
    <li>Tea</li>
    <li>Milk</li>
</ul>
```

Hacer los métodos necesarios en la clase usuario

Aplicación Nº 22 (Login)
Archivo: Login.php
método:POST

Recibe los datos del usuario(clave,mail)por POST ,
crear un objeto y utilizar sus métodos para poder verificar si es un usuario registrado, Retorna un :
“Verificado” si el usuario existe y coincide la clave también.
“Error en los datos” si esta mal la clave.
“Usuario no registrado si no coincide el mail“
Hacer los métodos necesarios en la clase usuario.